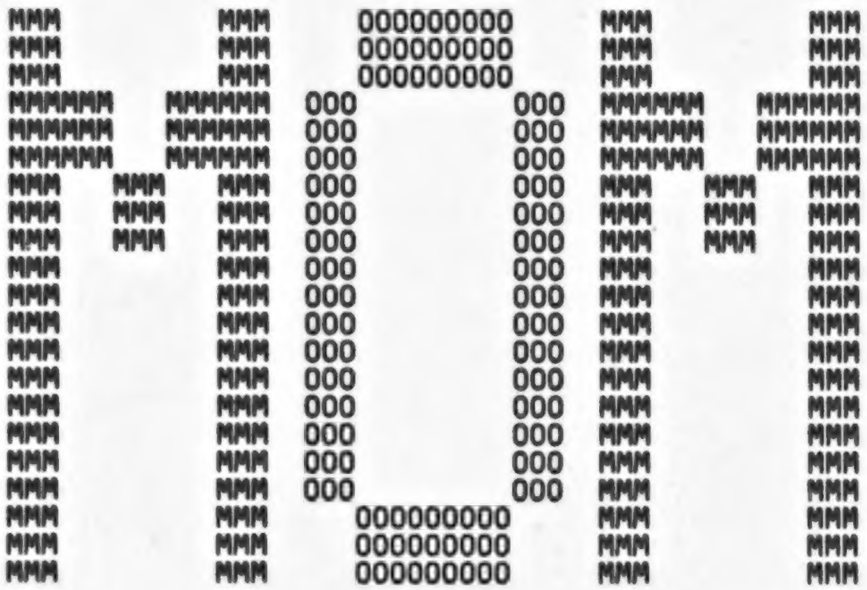
MMM MMM 000000000 MMM MMM
MMM MMM 000000000 MMM MMM
MMM MMM 000000000 MMM MMM
MMMMMMM MMMMMMM 000 000 MMMMMMM MMMMMM
MMMMMMM MMMMMMM 000 000 MMMMMMM MMMMMM
MMMMMMM MMMMMMM 000 000 MMMMMMM MMMMMM
MMM MMM MMM 000 000 MMM MMM MMM
MMM MMM MMM 000 000 MMM MMM MMM
MMM MMM MMM 000 000 MMM MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000 000 MMM MMM
MMM MMM 000000000 MMM MMM
MMM MMM 000000000 MMM MMM
MMM MMM 000000000 MMM MMM

FILEID**MOMPARSE

```
MM      MM    000000    MM      MM  PPPPPPPP      AAAAAA    RRRRRRRR      SSSSSSSS  EEEEEEEEEE
MM      MM    000000    MM      MM  PPPPPPPP      AAAAAA    RRRRRRRR      SSSSSSSS  EEEEEEEEEE
MMMM  MMMM  OO      OO  MMMM  MMMM  PP      PP  AA      AA  RR      RR  SS        EE
MMMM  MMMM  OO      OO  MMMM  MMMM  PP      PP  AA      AA  RR      RR  SS        EE
MM  MM  MM  OO      OO  MM  MM  MM  PP      PP  AA      AA  RR      RR  SS        EE
MM  MM  MM  OO      OO  MM  MM  MM  PP      PP  AA      AA  RR      RR  SS        EE
MM      MM  OO      OO  MM      MM  PPPPPPPP    AA      AA  RRRRRRRR      SSSSSS    EEEEEEEE
MM      MM  OO      OO  MM      MM  PPPPPPPP    AA      AA  RRRRRRRR      SSSSSS    EEEEEEEE
MM      MM  OO      OO  MM      MM  PP          AAAAAAAAAA  RR  RR              SS  EE
MM      MM  OO      OO  MM      MM  PP          AAAAAAAAAA  RR  RR              SS  EE
MM      MM  OO      OO  MM      MM  PP          AA      AA  RR    RR            SS  EE
MM      MM  OO      OO  MM      MM  PP          AA      AA  RR    RR            SS  EE
MM      MM    000000    MM      MM  PP          AA      AA  RR      RR  SSSSSSSS    EEEEEEEEEE
MM      MM    000000    MM      MM  PP          AA      AA  RR      RR  SSSSSSSS    EEEEEEEEEE

LL              IIIIII      SSSSSSSS
LL              IIIIII      SSSSSSSS
LL                II      SS
LL                II      SS
LL                II      SS
LL                II        SSSSSS
LL                II        SSSSSS
LL                II              SS
LL                II              SS
LL                II              SS
LL                II              SS
LLLLLLLLLL      IIIIII    SSSSSSSS
LLLLLLLLLL      IIIIII    SSSSSSSS
```

MOM
V04

```
;  1     0001  0
;  2     0002  0 %TITLE 'Maintenance Operations NPARSE action routines for parsing parameters'
;  3     0003  0 MODULE MOMPARSE (
;  4     0004  0                         LANGUAGE (BLISS32),
;  5     0005  0                         ADDRESSING_MODE (NONEXTERNAL=GENERAL),
;  6     0006  0                         ADDRESSING_MODE (EXTERNAL=GENERAL),
;  7     0007  0                         IDENT = 'V04-000'
;  8     0008  0                         ) =
;  9     0009  1 BEGIN
; 10     0010  1 !
; 11     0011  1 !*****************************************************************************
; 12     0012  1 !*                                                                           *
; 13     0013  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
; 14     0014  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
; 15     0015  1 !*  ALL RIGHTS RESERVED.                                                     *
; 16     0016  1 !*                                                                           *
; 17     0017  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 18     0018  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
; 19     0019  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
; 20     0020  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
; 21     0021  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
; 22     0022  1 !*  TRANSFERRED.                                                             *
; 23     0023  1 !*                                                                           *
; 24     0024  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
; 25     0025  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
; 26     0026  1 !*  CORPORATION.                                                             *
; 27     0027  1 !*                                                                           *
; 28     0028  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
; 29     0029  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
; 30     0030  1 !*                                                                           *
; 31     0031  1 !*                                                                           *
; 32     0032  1 !*****************************************************************************
; 33     0033  1 !
; 34     0034  1
; 35     0035  1 !++
; 36     0036  1 ! FACILITY:  DECnet-VAX V2.0 Network Management Listener
; 37     0037  1 !
; 38     0038  1 !
; 39     0039  1 ! ABSTRACT:
; 40     0040  1 !    This module contains action routines called by NPARSE to parse and
; 41     0041  1 !    store NICE entity parameters.
; 42     0042  1 !
; 43     0043  1 ! ENVIRONMENT:  VAX/VMS Operating System
; 44     0044  1 !
; 45     0045  1 ! AUTHOR:  Kathy Perko
; 46     0046  1 !
; 47     0047  1 ! CREATION DATE:  2-Jan-1983
; 48     0048  1 !
; 49     0049  1 ! MODIFIED BY:
; 50     0050  1 !       V03-005 MKP0005         Kathy Perko             13-July-1984
; 51     0051  1 !               Change NODE SERVICE PASSWORD from an H-8 field to an HI-8
; 52     0052  1 !               field.  The architecture conflicts with itself about it.
; 53     0053  1 !
; 54     0054  1 !       V03-004 MKP0004         Kathy Perko             6-June-1984
; 55     0055  1 !               Don't apply area 1 fix to exec.
; 56     0056  1 !
; 57     0057  1 !       V03-003 MKP0003         Kathy Perko             1-May-1984
```

```
;    58          0058  1 !                 Check for correct loop assistant parameter ID in LOOP
;    59          0059  1 !                 CIRCUIT parameter consistency check.
;    60          0060  1 !
;    61          0061  1 !       V03-002 MKP0002         Kathy Perko             28-Mar-1984
;    62          0062  1 !                 Fix area 1 problem.
;    63          0063  1 !
;    64          0064  1 !       V03-001 MKP0001         Kathy Perko             29-Jan-1984
;    65          0065  1 !                 Do some cross checking on LOOP CIRCUIT parameters.
;    66          0066  1 !                 Add a routine to check for loopback assist request messages,
;    67          0067  1 !                 and a routine to use the MOP message software ID field as
;    68          0068  1 !                 a load file ID.
;    69          0069  1 !
;    70          0070  1 !--
;    71          0071  1
```

```
;    73          0072  1 %SBTTL 'Declarations'
;    74          0073  1
;    75          0074  1 !
;    76          0075  1 ! TABLE OF CONTENTS:
;    77          0076  1 !
;    78          0077  1
;    79          0078  1 FORWARD ROUTINE
;    80          0079  1     mom$parse_nice_entity,
;    81          0080  1     mom$parse_function,
;    82          0081  1     mom$parse_option,
;    83          0082  1     mom$parse_entity_id,
;    84          0083  1     mom$save_param,
;    85          0084  1     mom$save_node_id,
;    86          0085  1     mom$check_node_entity,
;    87          0086  1     mom$check_loop_params,
;    88          0087  1     mom$mop_chk_loop_assist,
;    89          0088  1     mom$save_mop_msg,
;    90          0089  1     mom$save_load_file_id,
;    91          0090  1     mom_fix_node_num:   NOVALUE,
;    92          0091  1     mom$parse_error:    NOVALUE,
;    93          0092  1     mom$prsmoperr;
;    94          0093  1
;    95          0094  1 !
;    96          0095  1 ! INCLUDE FILES:
;    97          0096  1 !
;    98          0097  1
;    99          0098  1 LIBRARY 'LIB$:MOMLIB.L32';
;   100          0099  1 LIBRARY 'SHRLIB$:NMALIBRY.L32';
;   101          0100  1 LIBRARY 'SHRLIB$:NET.L32';
;   102          0101  1 LIBRARY 'SYS$LIBRARY:STARLET.L32';
;   103          0102  1
;   104          0103  1 !
;   105          0104  1 ! EXTERNAL REFERENCES:
;   106          0105  1 !
;   107          0106  1
;   108          0107  1 $mom_externals;                 ! Macro with common MOM externals.
;   109          0108  1
;   110          0109  1 EXTERNAL LITERAL
;   111          0110  1     mom$_badmopfct;
;   112          0111  1
;   113          0112  1 EXTERNAL
;   114          0113  1     mom$npa_init,               ! Nparse table for NICE message entities.
;   115          0114  1     mom$ab_ncp_version;
;   116          0115  1
;   117          0116  1 EXTERNAL ROUTINE
;   118          0117  1     nma$nparse,
;   119          0118  1     mom$build_p2,
;   120          0119  1     mom$netacp_qio,
;   121          0120  1     mom$error;
```

```
;   123     0121  1 %SBTTL 'mom$parse_nice_entity  Initial message parsing routine'
;   124     0122  1 GLOBAL ROUTINE mom$parse_nice_entity =
;   125     0123  1
;   126     0124  1 !++
;   127     0125  1 ! FUNCTIONAL DESCRIPTION:
;   128     0126  1 !       This routine invokes the NPARSE facility to check the funcition,
;   129     0127  1 !       option, and entity codes in a NICE request received from NCP.
;   130     0128  1 !
;   131     0129  1 ! IMPLICIT OUTPUTS:
;   132     0130  1 !
;   133     0131  1 !       MOM$GB_FUNCTION contains the function code.
;   134     0132  1 !       MOM$GB_OPTION_BYTE  contains the option codes.
;   135     0133  1 !       MOM$GL_ENTITY_CODE  contains the entity code.
;   136     0134  1 !       MOM$AB_NPARSE_BLK contains parsing information about the remainder
;   137     0135  1 !               of the message.
;   138     0136  1 !
;   139     0137  1 ! ROUTINE VALUE:
;   140     0138  1 ! COMPLETION CODES:
;   141     0139  1 !       If the parse fails then the error is signalled, and a NICE error
;   142     0140  1 !       response is built with the error specified by the parse state
;   143     0141  1 !       table. Otherwise success is returned.
;   144     0142  1 !
;   145     0143  1 !--
;   146     0144  1
;   147     0145  2 BEGIN
;   148     0146  2
;   149     0147  2 LOCAL
;   150     0148  2     status;                                 ! Temporary status
;   151     0149  2 !
;   152     0150  2 ! Initialize message parsing data
;   153     0151  2 !
;   154     0152  2 mom$gl_service_flags = 0;                   ! Clear internal options flags
;   155     0153  2 !
;   156     0154  2 ! Initialize the NPARSE argument block with the address and length
;   157     0155  2 ! of the NICE message to be parsed.  Then call the NPARSE facility
;   158     0156  2 ! to parse the function, option, and entity fields of the message.
;   159     0157  2 !
;   160     0158  2 mom$ab_nparse_blk [npa$l_msgptr] = mom$ab_nice_rcv_buf;
;   161     0159  2 mom$ab_nparse_blk [npa$l_msgcnt] = .mom$gl_nice_rcv_msg_len;
;   162     0160  2
;   163     0161  2 nma$nparse (mom$ab_nparse_blk, mom$npa_init);
;   164     0162  2 !
;   165     0163  2 ! If control returns here, the message parsed correctly.  Otherwise,
;   166     0164  2 ! an error was signalled and an error response returned to NCP via
;   167     0165  2 ! NML.
;   168     0166  2 !
;   169     0167  2 RETURN success
;   170     0168  2
;   171     0169  1 END;                                ! End of MOM$PARSE_NICE_ENTITY


;                                                               .TITLE  MOMPARSE Maintenance Operations NPARSE action r
                                                                        outines f
                                                                .IDENT  \V04-000\

                                                                .EXTRN  MOM$GL_LOGMASK, MOM$GL_SVD_INDEX
                                                                .EXTRN  MOM$AB_SERVICE_DATA
```

```
                                                .EXTRN  MOM$GB_FUNCTION
                                                .EXTRN  MOM$GB_OPTION_BYTE
                                                .EXTRN  MOM$GB_ENTITY_CODE
                                                .EXTRN  MOM$AB_ENTITY_BUF
                                                .EXTRN  MOM$GQ_ENTITY_BUF_DSC
                                                .EXTRN  MOM$GL_SERVICE_FLAGS
                                                .EXTRN  MOM$AB_NPARSE_BLK
                                                .EXTRN  MOM$AB_NICE_RCV_BUF
                                                .EXTRN  MOM$AB_NICE_XMIT_BUF
                                                .EXTRN  MOM$GQ_NICE_RCV_BUF_DSC
                                                .EXTRN  MOM$GL_NICE_RCV_MSG_LEN
                                                .EXTRN  MOM$GQ_NICE_XMIT_BUF_DSC
                                                .EXTRN  MOM$AB_MSGBLOCK
                                                .EXTRN  MOM$AB_ACPQIO_BUFFER
                                                .EXTRN  MOM$GQ_ACPQIO_BUF_DSC
                                                .EXTRN  MOM$AB_CIB, MOM$AB_LOOP_CIB
                                                .EXTRN  MOM$AB_TRIGGER_CIB
                                                .EXTRN  MOM$AB_MOP_XMIT_BUF
                                                .EXTRN  MOM$GQ_MOP_XMIT_BUF_DSC
                                                .EXTRN  MOM$AB_MOP_RCV_BUF
                                                .EXTRN  MOM$GQ_MOP_RCV_BUF_DSC
                                                .EXTRN  MOM$AB_MOP_MSG, MOM$GQ_MOP_MSG_DSC
                                                .EXTRN  MOM$GW_EVT_CODE
                                                .EXTRN  MOM$GB_EVT_POPR
                                                .EXTRN  MOM$GB_EVT_PRSN
                                                .EXTRN  MOM$GB_EVT_PSER
                                                .EXTRN  SVD$GK_PCNO_ADD
                                                .EXTRN  SVD$GK_PCNO_SDV
                                                .EXTRN  SVD$GK_PCNO_CPU
                                                .EXTRN  SVD$GK_PCNO_STY
                                                .EXTRN  SVD$GK_PCNO_DAD
                                                .EXTRN  SVD$GK_PCNO_DCT
                                                .EXTRN  SVD$GK_PCNO_IHO
                                                .EXTRN  SVD$GK_PCNO_NNA
                                                .EXTRN  SVD$GK_PCNO_SLI
                                                .EXTRN  SVD$GK_PCNO_SPA
                                                .EXTRN  SVD$GK_PCNO_HWA
                                                .EXTRN  SVD$GK_PCNO_SNV
                                                .EXTRN  SVD$GK_PCNO_LOA
                                                .EXTRN  SVD$GK_PCNO_SLO
                                                .EXTRN  SVD$GK_PCNO_TLO
                                                .EXTRN  SVD$GK_PCNO_DFL
                                                .EXTRN  SVD$GK_PCNO_SID
                                                .EXTRN  SVD$GK_PCNO_DUM
                                                .EXTRN  SVD$GK_PCNO_SDU
                                                .EXTRN  SVD$GK_PCNO_$HNA
                                                .EXTRN  SVD$GK_PCNO_$HHW
                                                .EXTRN  SVD$GK_PCNO_$FTY
                                                .EXTRN  SVD$GK_PCNO_PHA
                                                .EXTRN  SVD$GK_PCNO_$DA
                                                .EXTRN  SVD$GK_PCNO_LPC
                                                .EXTRN  SVD$GK_PCNO_LPL
                                                .EXTRN  SVD$GK_PCNO_LPD
                                                .EXTRN  SVD$GK_PCNO_LPH
                                                .EXTRN  SVD$GK_PCNO_LPA
                                                .EXTRN  SVD$GK_PCNO_LPN
                                                .EXTRN  SVD$GK_PCNO_$LNA
```

```
                                                                .EXTRN  SVD$GK_PCNO_$LNH
                                                                .EXTRN  SVD$GK_PCNO_LAN
                                                                .EXTRN  SVD$GK_PCNO_$LNN
                                                                .EXTRN  SVD$GK_PCNO_$LAH
                                                                .EXTRN  SVD$GK_PCLI_STI
                                                                .EXTRN  SVD$C_ENTRY_COUNT
                                                                .EXTRN  MOM$_BADMOPFCT, MOM$NPA_INIT
                                                                .EXTRN  MOM$AB_NCP_VERSION
                                                                .EXTRN  NMA$NPARSE, MOM$BUILD_P2
                                                                .EXTRN  MOM$NETACP_QIO, MOM$ERROR

                                                                .PSECT  $CODE$,NOWRT,2

                                                0004 00000      .ENTRY  MOM$PARSE_NICE_ENTITY, Save R2                  ; 0122
                                    52 00000000G   00   9E 00002 MOVAB   MOM$AB_NPARSE_BLK+8, R2                    ;
                                       00000000G   00   D4 00009 CLRL    MOM$GL_SERVICE_FLAGS                       ; 0152
                                    62 00000000G   00   9E 0000F MOVAB   MOM$AB_NICE_RCV_BUF, MOM$AB_NPARSE_BLK+8  ; 0158
                                FC  A2 00000000G   00   D0 00016 MOVL    MOM$GL_NICE_RCV_MSG_LEN, -                 ; 0159
                                                                        MOM$AB_NPARSE_BLK+4                         ;
                                       00000000G   00   9F 0001E PUSHAB  MOM$NPA_INIT                               ; 0161
                                                F8 A2   9F 00024 PUSHAB  MOM$AB_NPARSE_BLK                          ;
                                00000000G   00          02   FB 00027 CALLS   #2, NMA$NPARSE                        ;
                                            50          01   D0 0002E MOVL    #1, R0                                ; 0167
                                                        04   00031 RET                                             ; 0169

; Routine Size:  50 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  173          0170  1 %SBTTL 'mom$parse_function  Store function code (action routine)'
;  174          0171  1 GLOBAL ROUTINE mom$parse_function =
;  175          0172  1
;  176          0173  1 !++
;  177          0174  1 ! FUNCTIONAL DESCRIPTION:
;  178          0175  1 !
;  179          0176  1 !       Parse and store the function code from the NICE command message.
;  180          0177  1 !
;  181          0178  1 ! IMPLICIT OUTPUTS:
;  182          0179  1 !
;  183          0180  1 !       MOM$GB_FUNCTION contains the function code.
;  184          0181  1 !
;  185          0182  1 ! ROUTINE VALUE:
;  186          0183  1 ! COMPLETION CODES:
;  187          0184  1 !
;  188          0185  1 !       Always returns success (SS$_NORMAL).
;  189          0186  1 !
;  190          0187  1 !--
;  191          0188  1
;  192          0189  2 BEGIN
;  193          0190  2
;  194          0191  2 $npa_argdef;                        ! Define NPARSE block reference
;  195          0192  2
;  196          0193  2 mom$gb_function = .nparse_block [npa$b_byte]; ! Set function
;  197          0194  2
;  198          0195  2 RETURN ss$_normal
;  199          0196  2
;  200          0197  1 END;                                ! End of MOM$PARSE_FUNCTION


                                        0000 00000      .ENTRY  MOM$PARSE_FUNCTION, Save nothing                ; 0171
                    00000000G   00      18   AC   90 00002  MOVB    24(NPARSE_BLOCK), MOM$GB_FUNCTION          ; 0193
                                50           01   D0 0000A  MOVL    #1, R0                                     ; 0195
                                                  04 0000D  RET                                                ; 0197

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 0032
```

```
;   202          0198  1 %SBTTL 'mom$parse_option  Store NICE message option byte (action routine)'
;   203          0199  1 GLOBAL ROUTINE mom$parse_option =
;   204          0200  1
;   205          0201  1 !++
;   206          0202  1 ! FUNCTIONAL DESCRIPTION:
;   207          0203  1 !       This routine is a NPARSE action routine that is called while
;   208          0204  1 !       parsing a NICE message.  It saves the option byte in a global
;   209          0205  1 !       field.
;   210          0206  1 !
;   211          0207  1 ! IMPLICIT INPUTS:
;   212          0208  1 !       NPARSE_BLOCK [NPA$B_BYTE] contains the option byte.
;   213          0209  1 !
;   214          0210  1 ! ROUTINE VALUE:
;   215          0211  1 ! COMPLETION CODES:
;   216          0212  1 !       Success (SS$_NORMAL) is always returned.
;   217          0213  1 !
;   218          0214  1 !--
;   219          0215  1
;   220          0216  2 BEGIN
;   221          0217  2
;   222          0218  2 $npa_argdef;                        ! Define NPARSE block reference
;   223          0219  2 !
;   224          0220  2 ! Save the entity code from the NPARSE argument block
;   225          0221  2 !
;   226          0222  2 mom$gb_option_byte = .nparse_block [npa$b_byte];
;   227          0223  2
;   228          0224  2 RETURN ss$_normal
;   229          0225  2
;   230          0226  1 END;                                ! End of MOM$PARSE_OPTION


                                0000 00000          .ENTRY  MOM$PARSE_OPTION, Save nothing                  ; 0199
                    00000000G   00      18  AC  90 00002  MOVB    24(NPARSE_BLOCK), MOM$GB_OPTION_BYTE    ; 0222
                                50          01  D0 0000A  MOVL    #1, R0                                  ; 0224
                                            04 0000D    RET                                                 ; 0226

; Routine Size:  14 bytes,    Routine Base:  $CODE$ + 0040
```

```
;  232     0227  1 %SBTTL 'mom$parse_entity_id  Parse the service id'
;  233     0228  1 GLOBAL ROUTINE mom$parse_entity_id =
;  234     0229  1
;  235     0230  1 !++
;  236     0231  1 ! FUNCTIONAL DESCRIPTION:
;  237     0232  1 !       Parse the service id code from the MOP message or NICE command.
;  238     0233  1 !
;  239     0234  1 ! IMPLICIT INPUTS:
;  240     0235  1 !       NPARSE_BLOCK [NPA$L_PARAM] contains the MOM internal entity code
;  241     0236  1 !               (MOM$C_CIRCUIT, MOM$C_LINE, MOM$C_NODE, or MOM$C_NODEBYNAME).
;  242     0237  1 !
;  243     0238  1 ! OUTPUTS:
;  244     0239  1 !       MOM$AB_ENTITY_BUF contains the entity ID
;  245     0240  1 !       MOM$CQ_ENTITY_BUF_DSC contains a descriptor of the entity ID in
;  246     0241  1 !               MOM$AB_ENTITY_BUF.
;  247     0242  1 !       MOM$GB_ENTITY_CODE contains the MOM internal code for the entity.
;  248     0243  1 !--
;  249     0244  1
;  250     0245  2 BEGIN
;  251     0246  2
;  252     0247  2 $npa_argdef;
;  253     0248  2
;  254     0249  2 LOCAL
;  255     0250  2     adr,
;  256     0251  2     ent,
;  257     0252  2     len,
;  258     0253  2     svd_index;
;  259     0254  2
;  260     0255  2 ent = .nparse_block [npa$l_param];
;  261     0256  2 !
;  262     0257  2 ! Select parse table according to entity code.
;  263     0258  2 !
;  264     0259  2 SELECTU .ent OF
;  265     0260  2     SET
;  266     0261  2
;  267     0262  2     [mom$c_node]:
;  268     0263  3         BEGIN
;  269     0264  3         MAP
;  270     0265  3             adr: REF BBLOCK;
;  271     0266  3         len = 2;
;  272     0267  3         adr = .nparse_block [npa$l_fldptr];
;  273     0268  3         svd_index = svd$gk_pcno_add;
;  274     0269  3         !
;  275     0270  3         ! If the node area is 0 and it's not a Phase III (or less) NCP,
;  276     0271  3         ! change to area 1.
;  277     0272  3         !
;  278     0273  3         mom_fix_node_num (.adr);
;  279     0274  2         END;
;  280     0275  2
;  281     0276  2     [mom$c_line, mom$c_nodebyname]:
;  282     0277  3         BEGIN
;  283     0278  3         len = .(.nparse_block [npa$l_fldptr])<0,8>;
;  284     0279  3         adr = .nparse_block [npa$l_fldptr] + 1;
;  285     0280  3         IF .ent EQL mom$c_line THEN
;  286     0281  3             svd_index = svd$gk_pcno_sli
;  287     0282  3         ELSE
;  288     0283  3             svd_index = svd$gk_pcno_nna;
```

```
;  289     0284  2          END;
;  290     0285  2
;  291     0286  2      [mom$c_circuit]:
;  292     0287  3          BEGIN
;  293     0288  3          len = .(.nparse_block [npa$l_fldptr])<0,8>;
;  294     0289  3          adr = .nparse_block [npa$l_fldptr] + 1;
;  295     0290  3          svd_index = svd$gk_pcno_sli;
;  296     0291  2          END;
;  297     0292  2
;  298     0293  2      [ALWAYS]:
;  299     0294  3          BEGIN
;  300     0295  3          CH$MOVE (.len, .adr, mom$ab_entity_buf);
;  301     0296  3          mom$gq_entity_buf_dsc [0] = .len;
;  302     0297  3          !
;  303     0298  3          ! Put the entity ID into the Service Data Table so it will
;  304     0299  3          ! override the value returned from the volatile database.
;  305     0300  3          !
;  306     0301  3          mom$ab_service_data [.svd_index, svd$b_string_len] = .len;
;  307     0302  3          CH$MOVE (.len,
;  308     0303  3                   .adr,
;  309     0304  3                   mom$ab_service_data [.svd_index, svd$t_string]);
;  310     0305  3          mom$ab_service_data [.svd_index, svd$v_msg_param] = true;
;  311     0306  2          END;
;  312     0307  2
;  313     0308  2      TES;
;  314     0309  2  !
;  315     0310  2  ! Save the entity code.
;  316     0311  2  !
;  317     0312  2  mom$gb_entity_code = .ent;
;  318     0313  2
;  319     0314  2  RETURN ss$_normal
;  320     0315  2
;  321     0316  1  END;                                   ! End of mom$parse_entity_id


                                 0FFC 00000          .ENTRY  MOM$PARSE_ENTITY_ID, Save R2,R3,R4,R5,R6,-      ; 0228
                                                             R7,R8,R9,R10,R11
                      5B 00000000G   8F   D0 00002   MOVL    #SVD$GK_PCNO_SLI, R11                           ;
                      5A 00000000G   00   9E 00009   MOVAB   MOM$AB_SERVICE_DATA+8, R10                      ;
                      59       20    AC   D0 00010   MOVL    32(NPARSE_BLOCK), ENT                           ; 0255
                                     17   12 00014   BNEQ    1$                                              ; 0262
                      57             02   D0 00016   MOVL    #2, LEN                                         ; 0266
                      58       14    AC   D0 00019   MOVL    20(NPARSE_BLOCK), ADR                           ; 0267
                      56 00000000G   8F   D0 0001D   MOVL    #SVD$GK_PCNO_ADD, SVD_INDEX                     ; 0268
                                     58   DD 00024   PUSHL   ADR                                             ; 0273
            00000000V 00             01   FB 00026   CALLS   #1, MOM_FIX_NODE_NUM                           ;
                      01             59   D1 0002D 1$:  CMPL  ENT, #1                                       ; 0276
                                     05   13 00030   BEQL    2$                                              ;
                      03             59   D1 00032   CMPL    ENT, #3                                         ;
                                     1A   12 00035   BNEQ    4$                                              ;
                      57       14    BC   9A 00037 2$:  MOVZBL  @20(NPARSE_BLOCK), LEN                      ; 0278
58          14        AC             01   C1 0003B   ADDL3   #1, 20(NPARSE_BLOCK), ADR                      ; 0279
                      03             59   D1 00040   CMPL    ENT, #3                                         ; 0280
                                     05   12 00043   BNEQ    3$                                              ;
```

```
                                56                5B  D0 00045          MOVL    R11, SVD_INDEX                                  ; 0281
                                                  07  11 00048          BRB     4$
                                56 00000000G      8F  D0 0004A 3$:      MOVL    #SVD$GK_PCNO_NNA, SVD_INDEX                     ; 0283
                                02                59  D1 00051 4$:      CMPL    ENT, #2                                         ; 0286
                                                  0C  12 00054          BNEQ    5$                                              ;
                                57        14      BC  9A 00056          MOVZBL  @20(NPARSE_BLOCK), LEN                          ; 0288
                          58        14    AC      01  C1 0005A          ADDL3   #1, 20(NPARSE_BLOCK), ADR                       ; 0289
                                56                58  D0 0005F          MOVL    R11, SVD_INDEX                                  ; 0290
                00000000G 00              68      57  28 00062 5$:      MOVC3   LEN, (ADR), MOM$AB_ENTITY_BUF                   ; 0295
                          00000000G       00      57  D0 0006A          MOVL    LEN, MOM$GQ_ENTITY_BUF_DSC                      ; 0296
                                56 00000089       8F  C4 00071          MULL2   #137, R6                                        ; 0301
                              6A46                57  90 00078          MOVB    LEN, MOM$AB_SERVICE_DATA+8[R6]                  ;
                    01 AA46                   68  57  28 0007C          MOVC3   LEN, (ADR), MOM$AB_SERVICE_DATA+9[R6]           ; 0304
                          FF AA46             01  88 00082          BISB2   #1, MOM$AB_SERVICE_DATA+7[R6]                   ; 0305
                    00000000G                 00  59  90 00087          MOVB    ENT, MOM$GB_ENTITY_CODE                         ; 0312
                                50                01  D0 0008E          MOVL    #1, R0                                          ; 0314
                                                  04  00091          RET                                                     ; 0316

; Routine Size:  146 bytes,    Routine Base:  $CODE$ + 004E

;  322          0317  1
```

```
;   324     0318  1 %SBTTL 'mom$save_param  Save NICE parameter value'
;   325     0319  1 GLOBAL ROUTINE mom$save_param =
;   326     0320  1
;   327     0321  1 !++
;   328     0322  1 ! FUNCTIONAL DESCRIPTION:
;   329     0323  1 !       This is an NPARSE action routine that is called while parsing
;   330     0324  1 !       a NICE message from NCP or a MOP message from the target node.
;   331     0325  1 !       It saves a parameter in the Service Data Table and sets a flag
;   332     0326  1 !       to indicate that the parameter from the volatile database is
;   333     0327  1 !       not to be used for this operation (since one was supplied in
;   334     0328  1 !       the NICE or MOP message).
;   335     0329  1 !
;   336     0330  1 ! IMPLICIT INPUTS:
;   337     0331  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter data.
;   338     0332  1 !           NPA$L_FLDCNT is the parameter length.
;   339     0333  1 !           NPA$L_FLDPTR is a pointer to the parameter in the received
;   340     0334  1 !               message buffer.
;   341     0335  1 !       MOM$GL_SVD_INDEX contains the index into the Service Data table
;   342     0336  1 !           (MOM$AB_SERVICE_DATA).
;   343     0337  1 !
;   344     0338  1 ! IMPLICIT OUTPUTS:
;   345     0339  1 !       The parameter value or string is inserted into the Service Data Table.
;   346     0340  1 !
;   347     0341  1 ! ROUTINE VALUE:
;   348     0342  1 ! COMPLETION CODES:
;   349     0343  1 !       Always returns SS$_NORMAL.
;   350     0344  1 !
;   351     0345  1 !--
;   352     0346  1
;   353     0347  2 BEGIN
;   354     0348  2
;   355     0349  2 $npa_argdef;                    ! Define NPARSE block reference
;   356     0350  2
;   357     0351  2 LOCAL
;   358     0352  2     svd_index,                  ! Index into this parameter's entry in
;   359     0353  2                                 !       the Service Data table.
;   360     0354  2     msgsize,                    ! Resultant message size
;   361     0355  2     len,
;   362     0356  2     ptr;                        ! Temporary parameter pointer
;   363     0357  2 !
;   364     0358  2 ! Add descriptor entry for this parameter.
;   365     0359  2 !
;   366     0360  2 len = .nparse_block [npa$l_fldcnt];
;   367     0361  2 ptr = .nparse_block [npa$l_fldptr];
;   368     0362  2 !
;   369     0363  2 ! If the NPARSE tables specified a parameter, then it is the SVD (Service
;   370     0364  2 ! Data table) index.  This is true only when parsing MOP messages.  When
;   371     0365  2 ! parsing NICE messages, the SVD index must be saved when the parameter
;   372     0366  2 ! ID is parsed; this routine is not called until parsing reaches the
;   373     0367  2 ! parameter value.
;   374     0368  2 !
;   375     0369  2 IF .nparse_block [npa$l_param] NEQ 0 THEN
;   376     0370  2     svd_index = .nparse_block [npa$l_param]
;   377     0371  2 ELSE
;   378     0372  2     svd_index = .mom$gl_svd_index;
;   379     0373  2 !
;   380     0374  2 ! Save the parameter in the Service Data Table.
```

```
;  381          0375  2 !
;  382          0376  2 IF .mom$ab_service_data [.svd_index, svd$b_nice_type]
;  383          0377  2                                                   EQL svd$k_string THEN
;  384          0378  2     BEGIN
;  385          0379  2     len = .len - 1;
;  386          0380  2     CH$MOVE (.len, (.ptr + 1), mom$ab_service_data [.svd_index, svd$t_string]);
;  387          0381  2     mom$ab_service_data [.svd_index, svd$b_string_len] = .len;
;  388          0382  2     mom$ab_service_data [.svd_index, svd$v_msg_param] = true;
;  389          0383  2     END
;  390          0384  2 ELSE
;  391          0385  2     BEGIN
;  392          0386  2     !
;  393          0387  2     ! Save the parameter value.
;  394          0388  2     !
;  395          0389  2     CH$COPY (.len,
;  396          0390  2              .ptr,
;  397          0391  2              0,
;  398          0392  2              4,
;  399          0393  2              mom$ab_service_data [.svd_index, svd$l_param]);
;  400          0394  2     mom$ab_service_data [.svd_index, svd$v_msg_param] = true;
;  401          0395  2     END;
;  402          0396  2 !
;  403          0397  2 ! Clear SVD index because the parsing routines think they are simply
;  404          0398  2 ! setting a bit when they put the index into this variable.
;  405          0399  2 !
;  406          0400  2 mom$gl_svd_index = 0;
;  407          0401  2
;  408          0402  2 RETURN ss$_normal
;  409          0403  2
;  410          0404  1 END;                                ! End of MOM$SAVE_PARAM


                                        07FC 00000          .ENTRY  MOM$SAVE_PARAM, Save R2,R3,R4,R5,R6,R7,R8,- ;  0319
                                                                    R9,R10
                                    5A 00000000G   00   9E 00002       MOVAB   MOM$GL_SVD_INDEX, R10                ;
                                    59 00000000G   00   9E 00009       MOVAB   MOM$AB_SERVICE_DATA+9, R9            ;
                                    58        10   AC   D0 00010       MOVL    16(NPARSE_BLOCK), LEN                ;  0360
                                    51        14   AC   D0 00014       MOVL    20(NPARSE_BLOCK), PTR                ;  0361
                                              20   AC   D5 00018       TSTL    32(NPARSE_BLOCK)                     ;  0369
                                                   06   13 0001B       BEQL    1$                                   ;
                                    50        20   AC   D0 0001D       MOVL    32(NPARSE_BLOCK), SVD_INDEX          ;  0370
                                                   03   11 00021       BRB     2$                                   ;
                                    50             6A   D0 00023 1$:   MOVL    MOM$GL_SVD_INDEX, SVD_INDEX          ;  0372
                              56    50 00000089      8F   C5 00026 2$:   MULL3   #137, SVD_INDEX, R6                ;  0376
                              50    56               59   C1 0002E       ADDL3   R9, R6, R0                           ;  0380
                                    57        FE A946   9E 00032       MOVAB   MOM$AB_SERVICE_DATA+7[R6], R7        ;  0382
                                    03        FD A946   91 00037       CMPB    MOM$AB_SERVICE_DATA+6[R6], #3        ;  0377
                                                   0E   12 0003C       BNEQ    3$                                   ;
                                                   58   D7 0003E       DECL    LEN                                  ;  0379
                              60        01    A1       58   28 00040       MOVC3   LEN, 1(PTR), (R0)                    ;  0380
                                    FF A946             58   90 00045       MOVB    LEN, MOM$AB_SERVICE_DATA+8[R6]       ;  0381
                                                   06   11 0004A       BRB     4$                                   ;  0382
                        04        00        61       58   2C 0004C 3$:   MOVC5   LEN, (PTR), #0, #4, (R0)            ;  0393
                                                   60        00051
```

```
                                67              01  B8 00052 4$:      BISB2   #1, (R7)                                ; 0394
                                                6A  D4 00055          CLRL    MOM$GL_SVD_INDEX                        ; 0400
                                50              01  D0 00057          MOVL    #1, R0                                  ; 0402
                                                    04 0005A          RET                                             ; 0404

; Routine Size:  91 bytes,    Routine Base:  $CODE$ + 00E0
```

```
;  412     0405  1 %SBTTL 'mom$save_node_id        Save node id'
;  413     0406  1 GLOBAL ROUTINE mom$save_node_id =
;  414     0407  1
;  415     0408  1 !++
;  416     0409  1 ! FUNCTIONAL DESCRIPTION:
;  417     0410  1 !       This is an NPARSE action that saves a node id passed in
;  418     0411  1 !       a LOAD, TRIGGER, or LOOP command.
;  419     0412  1 !
;  420     0413  1 ! IMPLICIT INPUTS:
;  421     0414  1 !       NPARSE_BLOCK [NPA$L_FLDPTR] contains the pointer to the entity
;  422     0415  1 !           format code and id string.
;  423     0416  1 !
;  424     0417  1 ! ROUTINE VALUE:
;  425     0418  1 ! COMPLETION CODES:
;  426     0419  1 !       Always returns success (SS$_NORMAL).
;  427     0420  1 !
;  428     0421  1 !--
;  429     0422  1
;  430     0423  2 BEGIN
;  431     0424  2
;  432     0425  2 $npa_argdef;                    ! Define NPARSE block reference
;  433     0426  2
;  434     0427  2 LOCAL
;  435     0428  2     node_addr_svd,
;  436     0429  2     node_name_svd,
;  437     0430  2     length,
;  438     0431  2     addr;
;  439     0432  2 !
;  440     0433  2 ! The LOAD HOST parameter is a word rather than a node id (for which
;  441     0434  2 ! the node address is preceded by a byte of 0).
;  442     0435  2 !
;  443     0436  2 IF .nparse_block [npa$l_param] EQL mom$c_node_addr_param THEN
;  444     0437  2     BEGIN
;  445     0438  2     length = 0;
;  446     0439  2     addr = .nparse_block [npa$l_fldptr];
;  447     0440  2     END
;  448     0441  2 ELSE
;  449     0442  2     BEGIN
;  450     0443  2     !
;  451     0444  2     ! Get length and address of node id string.
;  452     0445  2     !
;  453     0446  2     length = .(.nparse_block [npa$l_fldptr])<0,8>; ! Get length
;  454     0447  2     addr = .nparse_block [npa$l_fldptr] + 1;
;  455     0448  2     END;
;  456     0449  2 SELECTONEU .mom$gl_svd_index OF
;  457     0450  2     SET
;  458     0451  2     [svd$gk_pcno_iho]:
;  459     0452  2         BEGIN
;  460     0453  2         node_addr_svd = svd$gk_pcno_iho;
;  461     0454  2         node_name_svd = svd$gk_pcno_$hna;
;  462     0455  2         END;
;  463     0456  2     [svd$gk_pcno_lpn]:
;  464     0457  2         BEGIN
;  465     0458  2         node_addr_svd = svd$gk_pcno_lpn;
;  466     0459  2         node_name_svd = svd$gk_pcno_$lna;
;  467     0460  2         END;
;  468     0461  2     [svd$gk_pcno_lan]:
```

```
;  469          0462  }          BEGIN
;  470          0463               node_addr_svd = svd$gk_pcno_lan;
;  471          0464               node_name_svd = svd$gk_pcno_$inn;
;  472          0465               END;
;  473          0466      TES;
;  474          0467  !
;  475          0468  ! If length is zero then id is a node address, otherwise it is a
;  476          0469  ! node name string.
;  477          0470  !
;  478          0471  IF .length EQL 0 THEN
;  479          0472      !
;  480          0473      ! Save the node address.
;  481          0474      !
;  482          0475      BEGIN
;  483          0476      BIND
;  484          0477          node_addr = mom$ab_service_data [.node_addr_svd, svd$l_param] :
;  485          0478                  BBLOCK;
;  486          0479      mom$ab_service_data [.node_addr_svd, svd$l_param] = .(.addr)<0,16>;
;  487          0480      mom$ab_service_data [.node_addr_svd, svd$v_msg_param] = true;
;  488          0481      !
;  489          0482      ! If the node area is 0 and it's not a Phase III (or less) NCP,
;  490          0483      ! change to area 1.
;  491          0484      !
;  492          0485      mom_fix_node_num (node_addr);
;  493          0486      END
;  494          0487  ELSE
;  495          0488      !
;  496          0489      ! If it's a node name, save it and get the node address from the
;  497          0490      ! volatile database.
;  498          0491      !
;  499          0492      BEGIN
;  500          0493      mom$ab_service_data [.node_name_svd, svd$b_string_len] = .length;
;  501          0494      CH$MOVE (.length, .addr,
;  502          0495               mom$ab_service_data [.node_name_svd, svd$t_string]);
;  503          0496      mom$ab_service_data [.node_name_svd, svd$v_msg_param] = true;
;  504          0497      END;
;  505          0498
;  506          0499  mom$gl_svd_index = 0;                    ! Reset parameter code
;  507          0500
;  508          0501  RETURN ss$_normal
;  509          0502
;  510          0503  1 END;                                ! End of mom$save_node_id


                                    0FFC 00000          .ENTRY  MOM$SAVE_NODE_ID, Save R2,R3,R4,R5,R6,R7,-      ; 0406
                                                                R8,R9,R10,R11
                5B 00000000G   00   9E 00002          MOVAB   MOM$GL_SVD_INDEX, R11                           ;
                5A 00000000G   8F   D0 00009          MOVL    #SVD$GK_PCNO_LAN, R10                           ;
                59 00000000G   8F   D0 00010          MOVL    #SVD$GK_PCNO_LPN, R9                            ;
                58 00000000G   8F   D0 00017          MOVL    #SVD$GK_PCNO_IHO, R8                            ;
                57 00000000G   00   9E 0001E          MOVAB   MOM$AB_SERVICE_DATA+9, R7                       ;
                01         20  AC   D1 00025          CMPL    32(NPARSE_BLOCK), #1                            ; 0436
                               08   12 00029          BNEQ    1$                                              ;
                               52   D4 0002B          CLRL    LENGTH                                          ; 0438
```

```
                        53          14   AC   D0 0002D        MOVL    20(NPARSE_BLOCK), ADDR                        ; 0439
                                         09   11 00031        BRB     2$                                            ; 0436
                        52          14   BC   9A 00033 1$:    MOVZBL  @20(NPARSE_BLOCK), LENGTH                     ; 0446
        53        14    AC               01   C1 00037        ADDL3   #1, 20(NPARSE_BLOCK), ADDR                    ; 0447
                        50               6B   D0 0003C 2$:    MOVL    MOM$GL_SVD_INDEX, R0                          ; 0449
                        58               50   D1 0003F        CMPL    R0, R8                                        ; 0451
                                         0C   12 00042        BNEQ    3$                                            ;
                        51               58   D0 00044        MOVL    R8, NODE_ADDR_SVD                             ; 0453
                        50 00000000G     8F   D0 00047        MOVL    #SVD$GK_PCNO_$HNA, NODE_NAME_SVD              ; 0454
                                         20   11 0004E        BRB     5$                                            ; 0449
                        59               50   D1 00050 3$:    CMPL    R0, R9                                        ; 0456
                                         0C   12 00053        BNEQ    4$                                            ;
                        51               59   D0 00055        MOVL    R9, NODE_ADDR_SVD                             ; 0458
                        50 00000000G     8F   D0 00058        MOVL    #SVD$GK_PCNO_$LNA, NODE_NAME_SVD              ; 0459
                                         0F   11 0005F        BRB     5$                                            ; 0449
                        5A               50   D1 00061 4$:    CMPL    R0, R10                                       ; 0461
                                         0A   12 00064        BNEQ    5$                                            ;
                        51               5A   D0 00066        MOVL    R10, NODE_ADDR_SVD                            ; 0463
                        50 00000000G     8F   D0 00069        MOVL    #SVD$GK_PCNO_$CNN, NODE_NAME_SVD              ; 0464
                                         52   D5 00070 5$:    TSTL    LENGTH                                        ; 0471
                                         1E   12 00072        BNEQ    6$                                            ;
                        51 00000089      8F   C4 00074        MULL2   #137, R1                                      ; 0477
        50              51               57   C1 0007B        ADDL3   R7, R1, R0                                    ;
                        60               63   3C 0007F        MOVZWL  (ADDR), (R0)                                  ; 0479
                  FE A741                01   88 00082        BISB2   #1, MOM$AB_SERVICE_DATA+7[R1]                 ; 0480
                                         50   DD 00087        PUSHL   R0                                            ; 0485
            00000000V   00               01   FB 00089        CALLS   #1, MOM_FIX_NODE_NUM                          ;
                                         17   11 00090        BRB     7$                                            ; 0471
        56              50 00000089      8F   C5 00092 6$:    MULL3   #137, NODE_NAME_SVD, R6                       ; 0493
                  FF A746                52   90 0009A        MOVB    LENGTH, MOM$AB_SERVICE_DATA+8[R6]             ;
      6746              63               52   28 0009F        MOVC3   LENGTH, (ADDR), MOM$AB_SERVICE_DATA+9[R6]     ; 0495
                  FE A746                01   88 000A4        BISB2   #1, MOM$AB_SERVICE_DATA+7[R6]                 ; 0496
                                         6B   D4 000A9 7$:    CLRL    MOM$GL_SVD_INDEX                              ; 0499
                        50               01   D0 000AB        MOVL    #1, R0                                        ; 0501
                                              04 000AE        RET                                                   ; 0503

; Routine Size:  175 bytes,    Routine Base:  $CODE$ + 013B
```

```
;  512          0504  1 %SBTTL  'mom$check_node_entity          Verify a node request'
;  513          0505  1 GLOBAL ROUTINE  mom$check_node_entity =
;  514          0506  1 !++
;  515          0507  1 ! FUNCTIONAL DESCRIPTION:
;  516          0508  1 !
;  517          0509  1 !       This is an NPARSE action routine that verifies the requested
;  518          0510  1 !       service request (LOAD/TRIGGER/DUMP) is a node request and
;  519          0511  1 !       not a circuit request.  The routine is called whenever a
;  520          0512  1 !       service request containing a service circuit is received.
;  521          0513  1 !
;  522          0514  1 ! IMPLICIT INPUTS:
;  523          0515  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter
;  524          0516  1 !       data.
;  525          0517  1 !
;  526          0518  1 !       MOM$GB_ENTITY_CODE contains the entity code which indicates if a
;  527          0519  1 !             circuit or node request.
;  528          0520  1 !
;  529          0521  1 ! ROUTINE VALUE:
;  530          0522  1 ! COMPLETION CODE:
;  531          0523  1 !       If request is a node request SUCCESS is returned.
;  532          0524  1 !       Otherwise a parameter not applicable error (NMA$C_STS_PNA) will
;  533          0525  1 !       be signalled.
;  534          0526  1 !
;  535          0527  1 ! SIDE EFFECTS:
;  536          0528  1 !       If error then message is signalled.
;  537          0529  1 !
;  538          0530  1 !--
;  539          0531  2 BEGIN
;  540          0532  2
;  541          0533  2 $npa_argdef;                        ! Define NPARSE block reference
;  542          0534  2 !
;  543          0535  2 ! Verify that request is not a circuit request (node request).
;  544          0536  2 ! Signal errror if circuit request.
;  545          0537  2 !
;  546          0538  2 IF .mom$gb_entity_code NEQ mom$c_node AND
;  547          0539  2    .mom$gb_entity_code NEQ mom$c_nodebyname THEN
;  548          0540  2     mom$error (nma$c_sts_pna,
;  549          0541  2                .(.nparse_block [npa$l_fldptr])<0,16>);
;  550          0542  2
;  551          0543  2 RETURN success
;  552          0544  1 END;                               ! End MOM$CHECK_NODE_ENTITY  routine


                                     0000 00000          .ENTRY  MOM$CHECK_NODE_ENTITY, Save nothing                ; 0505
                         50 00000000G  00   9A 00002     MOVZBL  MOM$GB_ENTITY_CODE, R0                             ; 0538
                                       13   13 00009     BEQL    1$                                                 ;
                         01            50   91 0000B     CMPB    R0, #1                                             ; 0539
                                       0E   13 0000E     BEQL    1$                                                 ;
                         7E       14   BC   3C 00010     MOVZWL  @20(NPARSE_BLOCK), -(SP)                           ; 0541
                         7E            16   CE 00014     MNEGL   #22, -(SP)                                         ; 0540
                 00000000G  00         02   FB 00017     CALLS   #2, MOM$ERROR                                      ;
                         50            01   D0 0001E 1$:  MOVL    #1, R0                                             ; 0543
                                             04 00021     RET                                                        ; 0544
```

; Routine Size: 34 bytes,   Routine Base: $CODE$ + 01EA

```
;   554          0545  1 %SBTTL  'mom$check_loop_params          Verify LOOP CIRCUIT parameters'
;   555          0546  1 GLOBAL ROUTINE  mom$check_loop_params =
;   556          0547  1 !++
;   557          0548  1 ! FUNCTIONAL DESCRIPTION:
;   558          0549  1 !
;   559          0550  1 !       This is an NPARSE action routine that verifies the requested
;   560          0551  1 !       LOOP CIRCUIT command does not contain contradictory or missing
;   561          0552  1 !       parameters.
;   562          0553  1 !
;   563          0554  1 ! IMPLICIT INPUTS:
;   564          0555  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter
;   565          0556  1 !       data.
;   566          0557  1 !       The Service Data table (SVD)
;   567          0558  1 !
;   568          0559  1 ! ROUTINE VALUE:
;   569          0560  1 ! COMPLETION CODE:
;   570          0561  1 !       If request OK, SUCCESS is returned.
;   571          0562  1 !       Otherwise a parameter missing error (NMA$C_STS_PMS) will
;   572          0563  1 !       be signalled.
;   573          0564  1 !
;   574          0565  1 ! SIDE EFFECTS:
;   575          0566  1 !       If error then message is signalled.
;   576          0567  1 !
;   577          0568  1 !--
;   578          0569  2 BEGIN
;   579          0570  2
;   580          0571  2 $npa_argdef;                    ! Define NPARSE block reference
;   581          0572  2 !
;   582          0573  2 ! If the LOOP CIRCUIT command specifies loop with assist and/or help type,
;   583          0574  2 ! it must be an Ethernet circuit, and therefore a PHYSICAL ADDRESS or NODE
;   584          0575  2 ! parameter must be specified.
;   585          0576  2 !
;   586          0577  3 IF (.mom$gl_service_flags [mom$v_loop_w_assist] OR
;   587          0578  2     .mom$ab_service_data [svd$gk_pcno_lph, svd$v_msg_param]) AND
;   588          0579  3    NOT (.mom$ab_service_data [svd$gk_pcno_pha, svd$v_msg_param] OR
;   589          0580  3         .mom$ab_service_data [svd$gk_pcno_lan, svd$v_msg_param] OR
;   590          0581  3         .mom$ab_service_data [svd$gk_pcno_$lna, svd$v_msg_param]) THEN
;   591          0582  2     mom$error (nma$c_sts_pms,
;   592          0583  2                nma$c_pcno_pha);
;   593          0584  2 !
;   594          0585  2 ! If the LOOP CIRCUIT command specifies LOOP HELP but no ASSISTANT
;   595          0586  2 ! PHYSICAL ADDRESS or NODE, return an error.
;   596          0587  2 !
;   597          0588  2 IF .mom$ab_service_data [svd$gk_pcno_lph, svd$v_msg_param] AND
;   598          0589  2    NOT .mom$gl_service_flags [mom$v_loop_w_assist] THEN
;   599          0590  2     mom$error (nma$c_sts_pms,
;   600          0591  2                nma$c_pcno_lpa);
;   601          0592  2
;   602          0593  2 RETURN success
;   603          0594  1 END;                            ! End MOM$CHECK_LOOP_PARAMS routine


                                 000C 00000          .ENTRY  MOM$CHECK_LOOP_PARAMS, Save R2,R3          ; 0546
                     53 00000000G   00   9E 00002    MOVAB   MOM$GL_SERVICE_FLAGS, R3                   ;
```

```
                        52 00000000G   00  9E 00009         MOVAB   MOM$ERROR, R2
                    07  63             03  E0 00010         BBS     #3, MOM$GL_SERVICE_FLAGS, 1$                    ;  0577
                        32 00000000*   00  E9 00014         BLBC    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPH*137>-    ;  0578
                                                                >+7>, 3$
                        16 00000000*   00  E8 0001B 1$:      BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_PHA*137>-    ;  0579
                                                                >+7>, 2$
                        0F 00000000*   00  E8 00022         BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LAN*137>-    ;  0580
                                                                >+7>, 2$
                        08 00000000*   00  E8 00029         BLBS    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_$LNA*-       ;  0581
                                                                137>>+7>, 2$
                                       0A  DD 00030         PUSHL   #10                                             ;  0582
                        7E             1D  CE 00032         MNEGL   #29, -(SP)
                        62             02  FB 00035         CALLS   #2, MOM$ERROR
                        0E 00000000*   00  E9 00038 2$:      BLBC    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_LPH*137>-    ;  0588
                                                                >+7>, 3$
                    0A  63             03  E0 0003F         BBS     #3, MOM$GL_SERVICE_FLAGS, 3$                    ;  0589
                        7E         99  8F  9A 00043         MOVZBL  #153, -(SP)                                     ;  0590
                        7E             1D  CE 00047         MNEGL   #29, -(SP)
                        62             02  FB 0004A         CALLS   #2, MOM$ERROR
                        50             01  D0 0004D 3$:      MOVL    #1, R0                                          ;  0593
                                           04 00050         RET                                                     ;  0594

; Routine Size:  81 bytes,    Routine Base:  $CODE$ + 020C
```

```
;   605          0595  1 %SBTTL  'mom$save_mop_msg                Save MOP message received from target'
;   606          0596  1 GLOBAL ROUTINE  mom$save_mop_msg =
;   607          0597  1 !++
;   608          0598  1 ! FUNCTIONAL DESCRIPTION:
;   609          0599  1 !
;   610          0600  1 !       This is an NPARSE action routine that is called when certain
;   611          0601  1 !       MOP messages are received from the target node.  These messages
;   612          0602  1 !       must be saved because, if the target does not receive a response
;   613          0603  1 !       within a certain time, the target retransmits them.  Therefore,
;   614          0604  1 !       MOM must be prepared to skip over retransmissions of the same
;   615          0605  1 !       message.  So, save the message here to do the comparison for
;   616          0606  1 !       retransmissions against.
;   617          0607  1 !
;   618          0608  1 ! IMPLICIT INPUTS:
;   619          0609  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter
;   620          0610  1 !       data.
;   621          0611  1 !
;   622          0612  1 ! ROUTINE VALUE:
;   623          0613  1 ! COMPLETION CODE:
;   624          0614  1 !
;   625          0615  1 ! SIDE EFFECTS:
;   626          0616  1 !       The MOP message and a descriptor of it are saved in MOM$AB_MOP_MSG
;   627          0617  1 !       and MOM$GQ_MOP_MSG_DSC respectively.
;   628          0618  1 !
;   629          0619  1 !--
;   630          0620  2 BEGIN
;    -1          0621  2
;   632          0622  2 $npa_argdef;                        ! Define NPARSE block reference
;   633          0623  2
;   634          0624  2 mom$gq_mop_msg_dsc [0] = .mom$ab_nparse_blk [npa$l_msgcnt];
;   635          0625  2 mom$gq_mop_msg_dsc [1] = mom$ab_mop_msg;
;   636          0626  2 CH$MOVE (.mom$ab_nparse_blk [npa$l_msgcnt],
;   637          0627  2           .mom$ab_nparse_blk [npa$l_msgptr],
;   638          0628  2           mom$ab_mop_msg);
;   639          0629  2 RETURN success
;   640          0630  1 END;                                ! End mom$save_mop_msg  routine


                                                 007C 00000          .ENTRY  MOM$SAVE_MOP_MSG, Save R2,R3,R4,R5,R6        ; 0596
                                    56 00000000G 00   9E 00002          MOVAB   MOM$AB_MOP_MSG, R6                           ;
                                    51 00000000G 00   D0 00009          MOVL    MOM$AB_NPARSE_BLK+4, R1                      ; 0624
                          00000000G 00               51   D0 00010          MOVL    R1, MOM$GQ_MOP_MSG_DSC                       ;
                          00000000G 00               66   9E 00017          MOVAB   MOM$AB_MOP_MSG, MOM$GQ_MOP_MSG_DSC+4         ; 0625
                                    50 00000000G 00   D0 0001E          MOVL    MOM$AB_NPARSE_BLK+8, R0                      ; 0627
                               66               60   51   28 00025          MOVC3   R1, (R0), MOM$AB_MOP_MSG                     ; 0626
                                    50           01   D0 00029          MOVL    #1, R0                                       ; 0629
                                                      04 0002C          RET                                                  ; 0630

; Routine Size:  45 bytes,    Routine Base:  $CODE$ + 025D
```

```
;  642          0631  1 %SBTTL  'mom$mop_chk_loop_assist        Check for MOP loop assist request'
;  643          0632  1 GLOBAL ROUTINE  mom$mop_chk_loop_assist =
;  644          0633  1 !++
;  645          0634  1 ! FUNCTIONAL DESCRIPTION:
;  646          0635  1 !
;  647          0636  1 !       This is an NPARSE action routine that is called during autoservice
;  648          0637  1 !       if a MOP messages is received which doesn't contain any of the
;  649          0638  1 !       recognized MOP function codes.  In this case, it could be a
;  650          0639  1 !       multicast request for loopback assistance on the Ethernet.  Check to
;  651          0640  1 !       make sure the message was sent to the cross company Loopback Assistance
;  652          0641  1 !       multicast address.  If so, return success so the volunteer assistance
;  653          0642  1 !       will be sent.
;  654          0643  1 !
;  655          0644  1 ! IMPLICIT INPUTS:
;  656          0645  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter
;  657          0646  1 !       data.
;  658          0647  1 !
;  659          0648  1 ! ROUTINE VALUE:
;  660          0649  1 ! COMPLETION CODE:
;  661          0650  1 !       Returns MOM$_SUC if the system sending the MOP message sent it
;  662          0651  1 !       to the NI multicast loopback assistance address.
;  663          0652  1 !
;  664          0653  1 !--
;  665          0654  2 BEGIN
;  666          0655  2
;  667          0656  2 LOCAL
;  668          0657  2     status;
;  669          0658  2
;  670          0659  2 BIND
;  671          0660  2     NI_loop_assis_mult = UPLIT (%X'000000CF', WORD (%X'0000'));
;  672          0661  2
;  673          0662  2 !
;  674          0663  2 ! Check to make sure the MOP message was sent to the multicast loopback
;  675          0664  2 ! assist address.  The destination address of the message was saved
;  676          0665  2 !
;  677          0666  2 status = mom$_badmopfct;
;  678          0667  2 IF .mom$gl_service_flags [mom$v_ni_circ] THEN
;  679          0668  3     BEGIN
;  680          0669  3     IF CH$EQL (mom$k_ni_addr_length, ni_loop_assis_mult,
;  681          0670  3         mom$k_ni_addr_length,
;  682          0671  3             mom$nb_service_data [svd$gk_pcno_$da, svd$t_string]) THEN
;  683          0672  3         status = success;
;  684          0673  2     END;
;  685          0674  2 RETURN .status;
;  686          0675  1 END;                                   ! End mom$mop_chk_loop_assist routine


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                                        000000CF  00000 P.AAA:  .LONG   207
                                            0000  00004        .WORD   0

                                                NI_LOOP_ASSIS_MULT= P.AAA

                                                        .PSECT  $CODE$,NOWRT,2
```

```
                                                  001C 00000          .ENTRY  MOM$MOP_CHK_LOOP_ASSIST, Save R2,R3,R4       ; 0632
                                  54 00000000G    8F   D0 00002          MOVL    #MOM$_BADMOPFCT, STATUS                      ; 0666
                              11 00000000G        00   01   E1 00009     BBC     #1, MOM$GL_SERVICE_FLAGS, 1$                 ; 0667
                    00000000*  00 00000000'       00   06   29 00011     CMPC3   #6, NI_LOOP_ASSIS_MULT, <-                   ; 0671
                                                                                  <MOM$AB_SERVICE_DATA+<$VD$GK_PCNO_$DA*137>>-;
                                                                                  +9>
                                                  03   12 0001D          BNEQ    1$                                           ;
                                  54              01   D0 0001F          MOVL    #1, STATUS                                   ; 0672
                                  50              54   D0 00022 1$:      MOVL    STATUS, R0                                   ; 0674
                                                       04 00025          RET                                                  ; 0675

; Routine Size:  38 bytes,    Routine Base:  $CODE$ + 028A
```

```
;  688          0676  1 %SBTTL  'mom$save_load_file_id    Save load file specification'
;  689          0677  1 GLOBAL ROUTINE  mom$save_load_file_id =
;  690          0678  1 !++
;  691          0679  1 ! FUNCTIONAL DESCRIPTION:
;  692          0680  1 !
;  693          0681  1 !       This is an NPARSE action routine MOM calls if it receives a
;  694          0682  1 !       MOP program load request which contains string in the software
;  695          0683  1 !       id field of the message.  Append the logical name MOM$LOAD
;  696          0684  1 !       to the string.  It will be translated by RMS when the load
;  697          0685  1 !       file is opened.  The logical name is used as security to make
;  698          0686  1 !       sure that only files in one directory can be loaded.
;  699          0687  1 !
;  700          0688  1 !
;  701          0689  1 ! IMPLICIT INPUTS:
;  702          0690  1 !       NPARSE_BLOCK (pointed to by AP) contains the parsed parameter
;  703          0691  1 !       data.
;  704          0692  1 !
;  705          0693  1 ! ROUTINE VALUE:
;  706          0694  1 ! COMPLETION CODE:
;  707          0695  1 !
;  708          0696  1 !--
;  709          0697  2 BEGIN
;  710          0698  2 $npa_argdef;                    ! Define NPARSE block reference
;  711          0699  2
;  712          0700  2 LOCAL
;  713          0701  2     file_svd,
;  714          0702  2     len
;  715          0703  2     MOP_ptr,
;  716          0704  2     svd_ptr;
;  717          0705  2
;  718          0706  2 !
;  719          0707  2 ! The software type field precedes the software ID in the MOP message.
;  720          0708  2 ! This field determines which load file (secondary, tertiary, or operating
;  721          0709  2 ! system) to load.  Put the load file id in the correct load file entry
;  722          0710  2 ! of the Service Data Table (SVD).
;  723          0711  2 !
;  724          0712  2 file_svd =
;  725          0713  3     (SELECTONEU .mom$ab_service_data [svd$gk_pcno_sty, svd$l_param] OF
;  726          0714  3         SET
;  727          0715  3         [nma$c_soft_terl]: svd$gk_pcno_tlo;     ! Tertiary loader
;  728          0716  3         [nma$c_soft_osys]: svd$gk_pcno_loa;     ! Operating system
;  729          0717  3         [OTHERWISE]: svd$gk_pcno_slo;           ! Secondary loader
;  730          0718  2         TES);
;  731          0719  2 !
;  732          0720  2 ! Concatenate the logical name, MOM$LOAD, with the file specification
;  733          0721  2 ! in the software ID field of the MOP message.
;  734          0722  2 !
;  735          0723  2 len = %CHARCOUNT ('MOM$LOAD:');
;  736          0724  2 svd_ptr = mom$ab_service_data [.file_svd, svd$t_string];
;  737          0725  2 svd_ptr = CH$MOVE (.len, UPLIT BYTE ('MOM$LOAD:'), .svd_ptr);
;  738          0726  2 !
;  739          0727  2 ! Save the software id in the Service Data Table.
;  740          0728  2 !
;  741          0729  2 len = .len + .nparse_block [npa$l_fldcnt] - 1;
;  742          0730  2 MOP_ptr = .nparse_block [npa$l_fldptr];
;  743          0731  2 CH$MOVE (.len, (.MOP_ptr + 1), .svd_ptr);
;  744          0732  2 mom$ab_service_data [.file_svd, svd$b_string_len] = .len;
```

```
;  745          0733  2 mom$ab_service_data [.file_svd, svd$v_msg_param] = true;
;  746          0734  2
;  747          0735  2 RETURN success;
;  748          0736  1 END;                                ! End mom$save_load_file_id routine


                                                        .PSECT  $PLIT$,NOWRT,NOEXE,2

                    3A  44  41  4F  4C  24  4D  4F  4D  00006 P.AAB:  .ASCII  \MOM$LOAD:\                                ;


                                                        .PSECT  $CODE$,NOWRT,2

                                                01FC 00000          .ENTRY  MOM$SAVE_LOAD_FILE_ID, Save R2,R3,R4,R5,R6,-; 0677
                                                                            R7,R8
                                58 00000000G    00   9E 00002             MOVAB   MOM$AB_SERVICE_DATA+9, R8
                                50 00000000*    00   D0 00009             MOVL    <<MOM$AB_SERVICE_DATA+<SVD$GK_PCNO_STY*137>-; 0713
                                                                            >+9>, R0
                                01                   50   D1 00010         CMPL    R0, #1                                  ; 0715
                                                     09   12 00013         BNEQ    1$
                                50 00000000G         8F   D0 00015         MOVL    #SVD$GK_PCNO_TLO, FILE_SVD
                                                     15   11 0001C         BRB     3$
                                02                   50   D1 0001E 1$:     CMPL    R0, #2                                  ; 0716
                                                     09   12 00021         BNEQ    2$
                                50 00000000G         8F   D0 00023         MOVL    #SVD$GK_PCNO_LOA, FILE_SVD
                                                     07   11 0002A         BRB     3$
                                50 00000000G         8F   D0 0002C 2$:     MOVL    #SVD$GK_PCNO_SLO, FILE_SVD              ; 0717
                                56                   09   D0 00033 3$:     MOVL    #9, LEN                                 ; 0723
                    57          50 00000089          8F   C5 00036         MULL3   #137, FILE_SVD, R7                      ; 0724
                    53          57                   58   C1 0003E         ADDL3   R8, R7, SVD_PTR
                    63 00000000'    00               56   28 00042         MOVC3   LEN, P.AAB, (SVD_PTR)                   ; 0725
                    50              56         10    AC   C1 0004A         ADDL3   16(NPARSE_BLOCK), LEN, R0               ; 0729
                                    56         FF    A0   9E 0004F         MOVAB   -1(R0), LEN
                                    50         14    AC   D0 00053         MOVL    20(NPARSE_BLOCK), MOP_PTR               ; 0730
                                63      01         A0   28 00057         MOVC3   LEN, 1(MOP_PTR), (SVD_PTR)              ; 0731
                                    FF A847            56   90 0005C         MOVB    LEN, MOM$AB_SERVICE_DATA+8[R7]          ; 0732
                                    FE A847            01   88 00061         BISB2   #1, MOM$AB_SERVICE_DATA+7[R7]           ; 0733
                                        50             01   D0 00066         MOVL    #1, R0                                  ; 0735
                                                          04 00069          RET                                             ; 0736

; Routine Size:  106 bytes,    Routine Base:  $CODE$ + 02B0
```

```
;  750     0737  1 %SBTTL 'MOM_FIX_NODE_NUM    Fix node address parameter (action routine)'
;  751     0738  1 ROUTINE MOM_FIX_NODE_NUM (NODE_ADDR) : NOVALUE =
;  752     0739  1
;  753     0740  1 !++
;  754     0741  1 ! FUNCTIONAL DESCRIPTION:
;  755     0742  1 !
;  756     0743  1 !       This is an parsing action routine that checks the node address.  If
;  757     0744  1 !       the area number is 0 it can be one of two cases:
;  758     0745  1 !               The NCP is a Phase IV NCP, and user did not specify an area
;  759     0746  1 !               number in the NCP command.  In this case, assume the user
;  760     0747  1 !               means area 1 (since 0 is an invalid area number).
;  761     0748  1 !
;  762     0749  1 !               the NCP is a Phase III NCP and therefore doesn't understand
;  763     0750  1 !               area numbers.  In this case, assume the user means the
;  764     0751  1 !               executor node's area.
;  765     0752  1 !
;  766     0753  1 ! FORMAL PARAMETERS:
;  767     0754  1 !       NODE_ADDR       Address of Node address to fix.
;  768     0755  1 !
;  769     0756  1 ! IMPLICIT INPUTS:
;  770     0757  1 !       None
;  771     0758  1 !
;  772     0759  1 ! IMPLICIT OUTPUTS:
;  773     0760  1 !       None
;  774     0761  1 !
;  775     0762  1 !--
;  776     0763  1
;  777     0764  2 BEGIN
;  778     0765  2
;  779     0766  2 MAP
;  780     0767  2     node_addr    : REF BBLOCK [2];
;  781     0768  2
;  782     0769  2 LOCAL
;  783     0770  2     exec_addr    : BBLOCK [2];
;  784     0771  2
;  785     0772  2 !
;  786     0773  2 ! If the node address is 0, then it's the executor, so leave it that way.
;  787     0774  2 ! If the area number of the address is 0, then change it.
;  788     0775  2 !
;  789     0776  2 IF .node_addr [nma$v_addr] NEQ 0 AND
;  790     0777  2    .node_addr [nma$v_area] EQL 0 THEN
;  791     0778  2     BEGIN
;  792     0779  3     !
;  793     0780  3     ! If NCP is a Phase III NCP, use area 0 for the volatile database.
;  794     0781  3     ! NETACP will assume the executor's area number.  For permanent database,
;  795     0782  3     ! get the exec address from the permanent database and use it's area number.
;  796     0783  3     !
;  797     0784  3     IF CH$RCHAR (mom$ab_ncp_version) LEQ 3 then
;  798     0785  3         node_addr [nma$v_area] = 0
;  799     0786  3     ELSE
;  800     0787  3     !
;  801     0788  3     ! If NCP is a Phase IV NCP, use area 1.
;  802     0789  3     !
;  803     0790  3         node_addr [nma$v_area] = 1;
;  804     0791  3     END;
;  805     0792  2 RETURN
;  806     0793  2
```

```
;  807          0794  1 END;                          ! End of MOM_FIX_NODE_NUM



                                                0000 00000 MOM_FIX_NODE_NUM:
                                                            .WORD   Save nothing                                    ; 0738
                          50        04          AC  D0 00002          MOVL    NODE_ADDR, R0                       ; 0776
                      03FF 8F                   60  B3 00006          BITW    (R0), #1023                         :
                                                1B  13 0000B          BEQL    2$                                  :
                        FC 8F       01          A0  93 0000D          BITB    1(R0), #252                         ; 0777
                                                14  12 00012          BNEQ    2$                                  :
                           03 00000000G         00  91 00014          CMPB    MOM$AB_NCP_VERSION, #3              ; 0784
                                                06  1A 0001B          BGTRU   1$                                  :
                        01  A0          FC      8F  8A 0001D          BICB2   #252, 1(R0)                         ; 0785
                                                04  00022          RET                                         :
                60              06          0A          01  F0 00023 1$:    INSV    #1, #10, #6, (R0)           ; 0790
                                                04  00028 2$:    RET                                         ; 0794

; Routine Size:  41 bytes,    Routine Base:  $CODE$ + 031A
```

```
1 %SBTTL 'mom$parse_error  Build and signal error (action routine)'
1 GLOBAL ROUTINE mom$parse_error : NOVALUE =
1
1 !++
1 ! FUNCTIONAL DESCRIPTION:
1 !       This NPARSE action routine is called if an error is found when parsing
1 !       the parameters of a NICE command message.  It signals the error.
1 !
1 ! FORMAL PARAMETERS:
1 !       NONE
1 !
1 ! IMPLICIT INPUTS:
1 !       NPARSE argument block.
1 !               NPA$L_PARAM contains the error code.
1 !               NPA$L_FLDPTR points to the parameter in the message.
1 !
1 ! SIDE EFFECTS:
1 !       Error message is signalled.
1 !
1 !--
1
2 BEGIN
2
2 $npa_argdef;                                  ! Define NPARSE block reference
2
2 LOCAL
2     err_code,                                 ! Error code
2     err_detail;                               ! Error detail
2
2 err_code = .nparse_block [npa$l_param]; ! Get error code
2 !
2 ! Check for parameters to move in addition to error status.
2 !
3 err_detail = (
3     SELECTONEU .err_code OF
3         SET
3         [nma$c_sts_pty,
3         nma$c_sts_pva,
3         nma$c_sts_pna]:
3             .(.nparse_block [npa$l_msgptr] - 2)<0,16,0>; ! Get detail code
3
3         [OTHERWISE]:
3             -1;
3
2         TES);
2
2 mom$error (.err_code, .err_detail);           ! Signal error message
2
1 END;                                  ! End of MOM$PARSE_ERROR



                                 0000 00000          .ENTRY  MOM$PARSE_ERROR, Save nothing               ; 0796
                51       20   AC   D0 00002          MOVL    32(NPARSE_BLOCK), ERR_CODE                  ; 0824
      FFFFFFEA  8F            51   D1 00006          CMPL    ERR_CODE, #-22                              ; 0831
```

```
                                 12  13 0000D         BEQL    1$                                  ;
       FFFFFFF0  8F              51  D1 0000F         CMPL    ERR_CODE, #-16                      ;
                                 09  13 00016         BEQL    1$                                  ;
       FFFFFFFA  8F              51  D1 00018         CMPL    ERR_CODE, #-6                       ;
                                 0A  12 0001F         BNEQ    2$                                  ;
                 50       08     AC  D0 00021 1$:     MOVL    8(NPARSE_BLOCK), R0                 ; 0834
                 50       FE     A0  3C 00025         MOVZWL  -2(R0), ERR_DETAIL                  ;
                                 03  11 00029         BRB     3$                                  ;
                 50              01  CE 0002B 2$:     MNEGL   #1, ERR_DETAIL                      ; 0837
                                 50  DD 0002E 3$:     PUSHL   ERR_DETAIL                          ; 0841
                                 51  DD 00030         PUSHL   ERR_CODE                            ;
       00000000G 00              02  FB 00032         CALLS   #2, MOM$ERROR                       ;
                                     04 00039         RET                                         ; 0843

; Routine Size:  58 bytes,    Routine Base:  $CODE$ + 0343
```

```
;  859          0844  1 %SBTTL 'mom$prsmoperr  MOP parameter parsing error'
;  860          0845  1 GLOBAL ROUTINE mom$prsmoperr =
;  861          0846  1
;  862          0847  1 !++
;  863          0848  1 ! FUNCTIONAL DESCRIPTION:
;  864          0849  1 !
;  865          0850  1 !       This routine sets up response message information for errors
;  866          0851  1 !       encountered in parsing MOP messages.
;  867          0852  1 !
;  868          0853  1 ! FORMAL PARAMETERS:
;  869          0854  1 !
;  870          0855  1 !       NONE
;  871          0856  1 !
;  872          0857  1 ! IMPLICIT INPUTS:
;  873          0858  1 !
;  874          0859  1 !       The NPARSE argument block (NPA$L_PARAM) contains the code for
;  875          0860  1 !       the optional text message to be signalled.
;  876          0861  1 !
;  877          0862  1 ! IMPLICIT OUTPUTS:
;  878          0863  1 !
;  879          0864  1 !       MOM$AB_MSGBLOCK contains the response message information.
;  880          0865  1 !
;  881          0866  1 !--
;  882          0867  1
;  883          0868  2 BEGIN
;  884          0869  2
;  885          0870  2 $npa_argdef;
;  886          0871  2
;  887          0872  2 !
;  888          0873  2 ! Set up MOP protocol error with optional text message.
;  889          0874  2 !
;  890          0875  2 mom$ab_msgblock [msb$l_flags] = msb$m_msg_fld;
;  891          0876  2 mom$ab_msgblock [msb$b_code]  = nma$c_sts_lpr;
;  892          0877  2 mom$ab_msgblock [msb$l_text]  = .nparse_block [npa$l_param];
;  893          0878  2
;  894          0879  2 RETURN success
;  895          0880  2
;  896          0881  1 END;                                ! End of mom$prsmoperr



                                        0004 00000      .ENTRY  MOM$PRSMOPERR, Save R2                                  ; 0845
                    52 00000000G    00  9E 00002      MOVAB   MOM$AB_MSGBLOCK, R2                                     ;
                    62              04  D0 00009      MOVL    #4, MOM$AB_MSGBLOCK                                     ; 0875
                04  A2              11  8E 0000C      MNEGB   #17, MOM$AB_MSGBLOCK+4                                  ; 0876
                0C  A2          20  AC  D0 00010      MOVL    32(NPARSE_BLOCK), MOM$AB_MSGBLOCK+12                    ; 0877
                    50              01  D0 00015      MOVL    #1, R0                                                  ; 0879
                                    04  00018      RET                                                             ; 0881

; Routine Size:  25 bytes,    Routine Base:  $CODE$ + 037D

;  897          0882  1
;  898          0883  1
;  899          0884  1
```

```
;  900          0885  1 END                                           ! End of module
;  901          0886  1
;  902          0887  0 ELUDOM
```

```
;                                  PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  $CODE$                          918  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  $PLIT$                           15  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
```

```
;                               Library Statistics
;                                       --------  Symbols  --------      Pages       Processing
;       File                            Total     Loaded   Percent       Mapped      Time
;
;  _$255$DUA28:[MOM.OBJ]MOMLIB.L32;1      194         33        17           21        00:00.1
;  _$255$DUA28:[SHRLIB]NMALIBRY.L32;1     887         11         1           47        00:00.2
;  _$255$DUA28:[SHRLIB]NET.L32;1         1279          0         0           63        00:00.3
;  _$255$DUA28:[SYSLIB]STARLET.L32;1     9776          1         0          581        00:03.1
```

```
;                                   COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:MOMPARSE/OBJ=OBJ$:MOMPARSE MSRC$:MOMPARSE/UPDATE=(ENH$:MOMPARSE)

; Size:          918 code + 15 data bytes
; Run Time:          00:21.5
; Elapsed Time:      00:48.7
; Lines/CPU Min:      2471
; Lexemes/CPU-Min:  12398
; Memory Used:   107 pages
; Compilation Complete
```

MOMMSG
LIS

MOMMAIN
LIS

MOMPARSE
LIS

MOMSERSTA
LIS

MOMMOPSTA
LIS

MOMSERVIC
LIS

MOMMOPLIO
LIS

MOMTESSTA
LIS

MOMRSXDEF
LIS

MOMSUBS
LIS